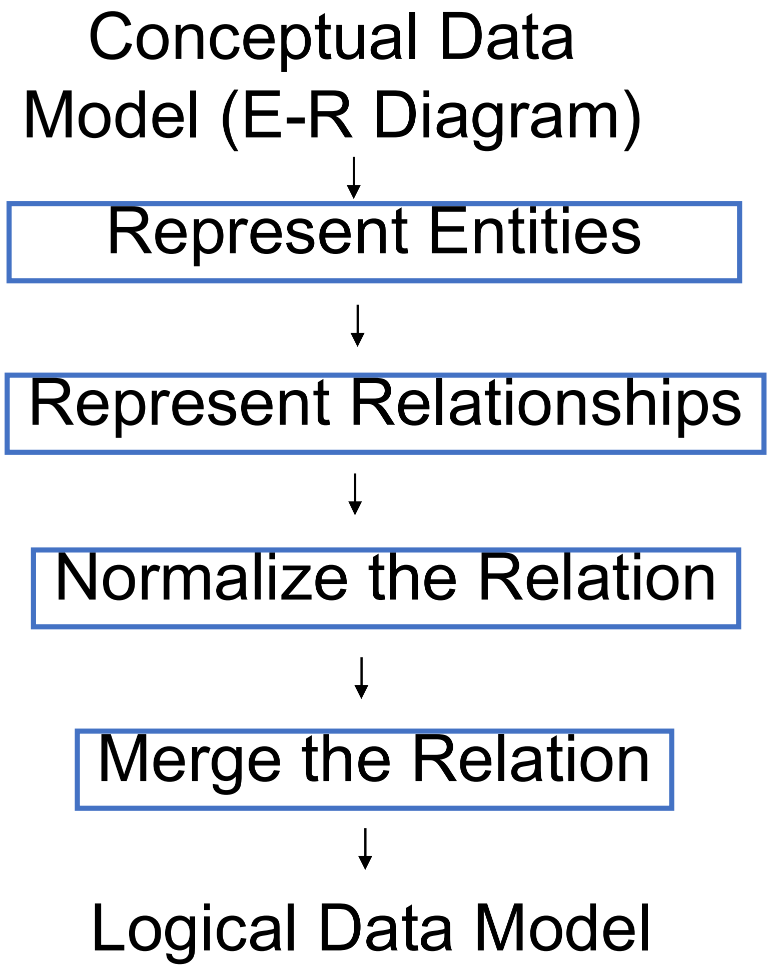
Chapter 4

Logical Database Design & the Relational Model

4 Steps in Logical Design Process



Conceptual Data Model (E-R Diagram)

Represent Entities

Represent Relationships

Normalize the Relation

Merge the Relation

Logical Data Model

4 Steps in Logical Design Process

- Represent Entities: Each entity in an E-R diagram is represented as a relation (a named two-dimensional table of data)
- Represent relationships: Various methods…Ex: Primary key of a relation is a foreign key of another (or) create a separate relation representing the relationship

4 Steps in Logical Design Process

- Normalize the relations: redesigning for well structured relations that avoid anomalies
- Merge the relations: Delete redundant relations

Relation

- A named two-dimensional table of data. Each relation consists of a finite set of named columns and an arbitrary number of unnamed rows.
- Notation:
  - RELATION NAME(Attribute, Attribute…)
- Example:
  - EMPLOYEE1(EmpID, Name, Dept, Salary)

Correspondence with ER Model

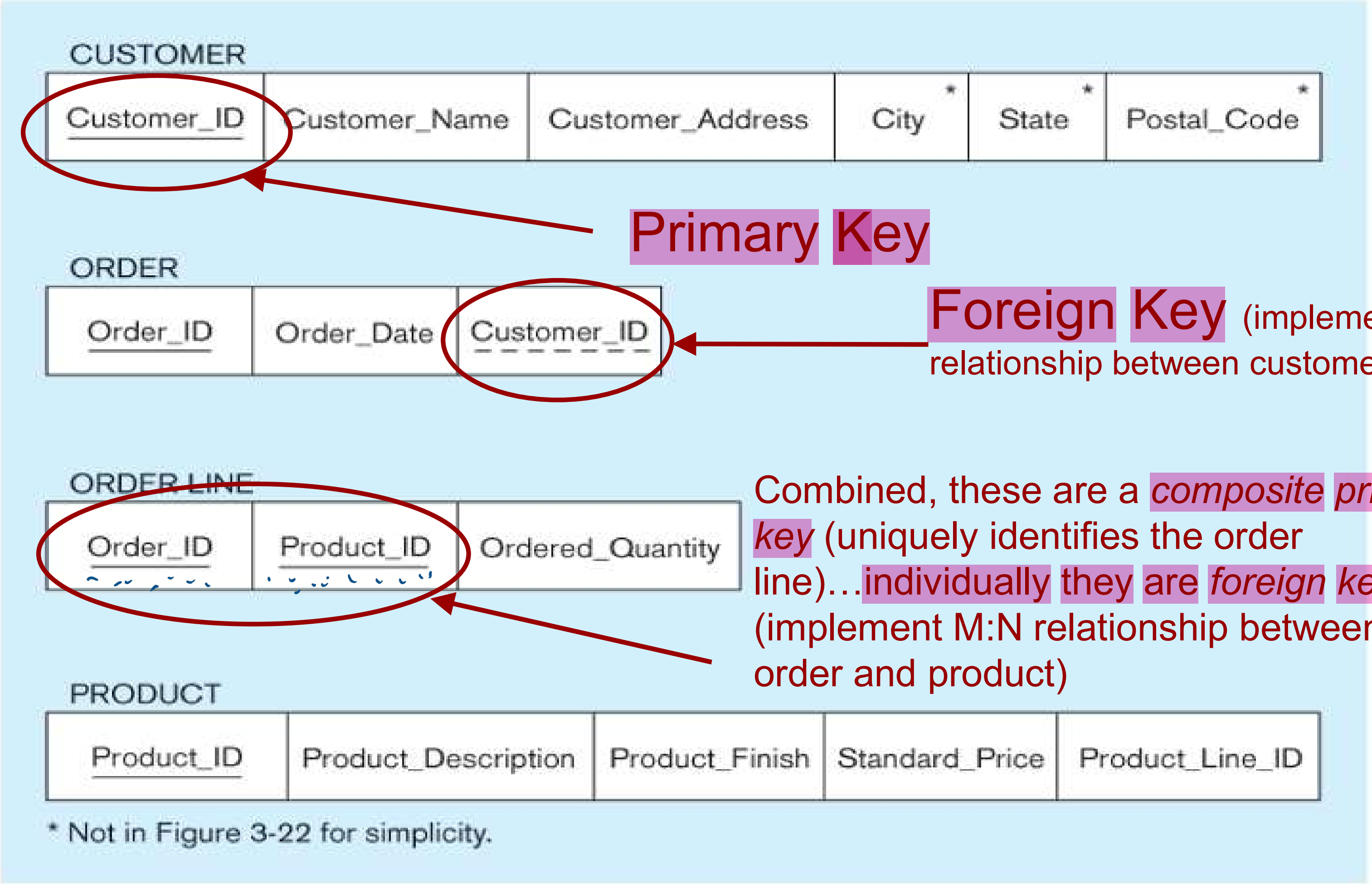
- Relations (tables) correspond with entity types
- Rows correspond with entity instances
- Columns correspond with attributes

- NOTE: The word *relation* (in relational database) is NOT the same same the word *relationship* (in ER model)

Relational Keys

- Allow storage and retrieval of a row of data based on stored values of that data
- Primary Key: An attribute (or combination of attributes) that uniquely identifies each row in a relation
- Composite Key: A primary key that consists of more than one attribute
- Foreign Key: An attribute that appears as a non-key attribute in one relation and as a primary key attribute (or part of a primary key) in another relation.

Figure 5-3 Schema for four relations (Pine Valley Furniture Company)



CUSTOMER

Customer_ID	Customer_Name	Customer_Address	City*	State*	Postal_Code*

ORDER

Order_ID	Order_Date	Customer_ID

ORDER LINE

Order_ID	Product_ID	Ordered_Quantity

PRODUCT

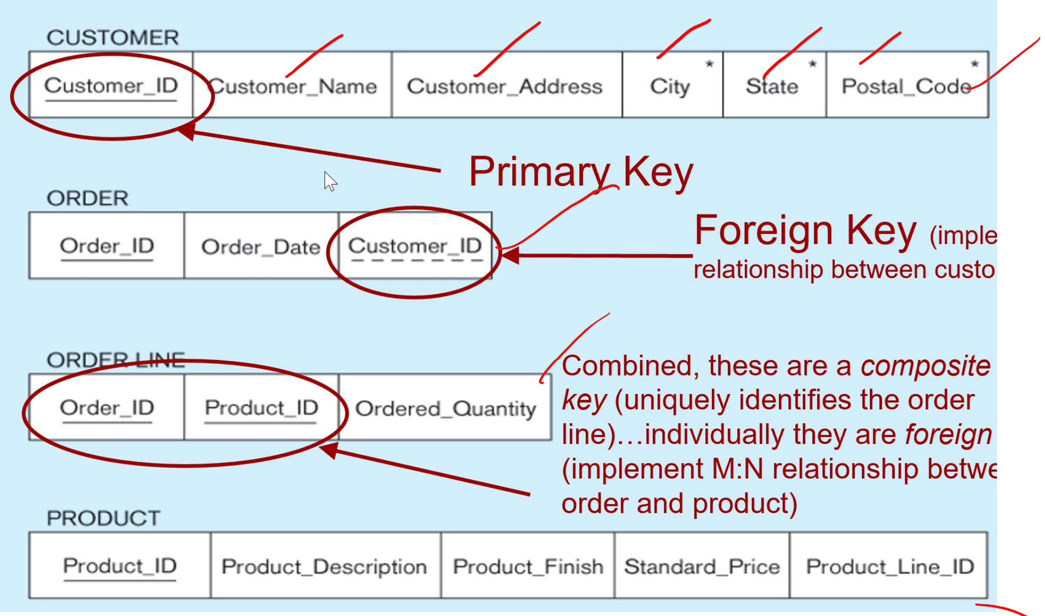
Product_ID	Product_Description	Product_Finish	Standard_Price	Product_Line_ID

* Not in Figure 3-22 for simplicity.

Primary Key

Foreign Key (implements 1:N relationship between customer and order)

Combined, these are a *composite primary key* (uniquely identifies the order line)…individually they are *foreign keys* (implement M:N relationship between order and product)



CUSTOMER

Customer_ID	Customer_Name	Customer_Address	City *	State *	Postal_Code *

Primary Key

ORDER

Order_ID	Order_Date	Customer_ID

Foreign Key (imple
relationship between custo

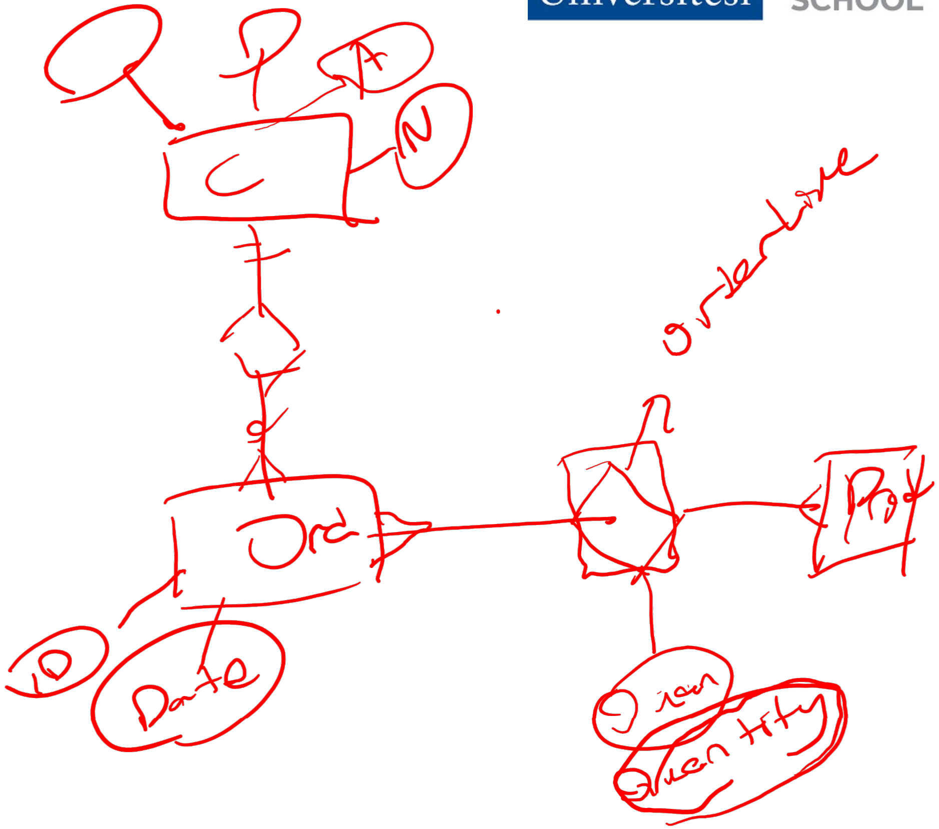
ORDER LINE

Order_ID	Product_ID	Ordered_Quantity

Combined, these are a *composite key* (uniquely identifies the order line)…individually they are *foreign* (implement M:N relationship betwe order and product)

PRODUCT

Product_ID	Product_Description	Product_Finish	Standard_Price	Product_Line_ID



assoc.
entity.

Properties of Relations

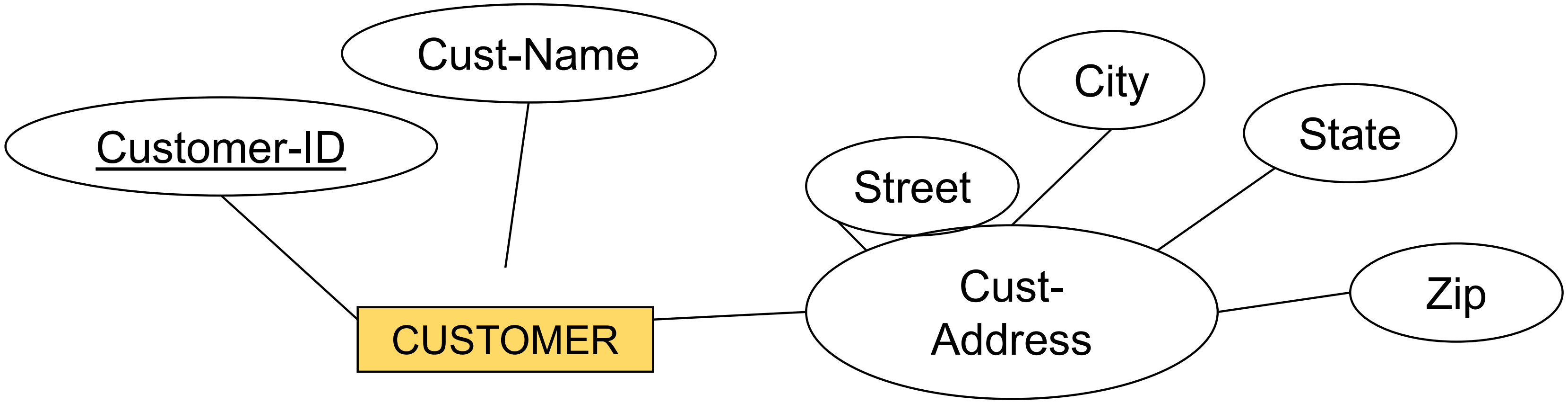
- Each relation (or table) has a unique name
- Entries in columns are atomic (no repeating groups - single valued)
- Entries in columns are from the same domain (formatted the same)
- Each row is unique (no duplicate rows)
- The sequence of columns (left to right) is insignificant
- The sequence of rows (top to bottom) is insignificant

Domain type : integer,string, long etc.
Her bir kolon altndaki instance(row) için domain ayn olmal
Her bir kolon altndaki instancelarda repeating group olmaml unique olmal.

Transforming EER Diagrams into Relations

- Regular Entities
  - Name of relation is name of Entity
  - Each simple attribute of entity becomes an attribute or column of relation
  - Identifier of entity becomes primary key of relation
  - Composite attributes: only simple components are included in relation

1. Mapping Regular Entity

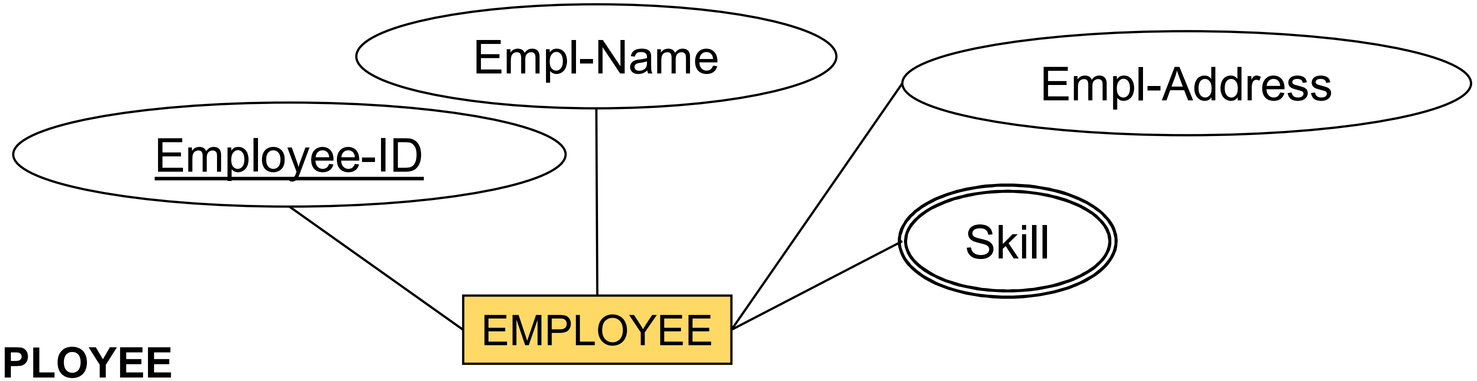


CUSTOMER

<u>Customer-ID</u>	Cust-Name	Street	City	State	Zip

Mapping Multi valued Attributes

- When a regular entity contains multivalued attributes, two relations are created



EMPLOYEE

Employee-ID	Empl-Name	Empl-Address

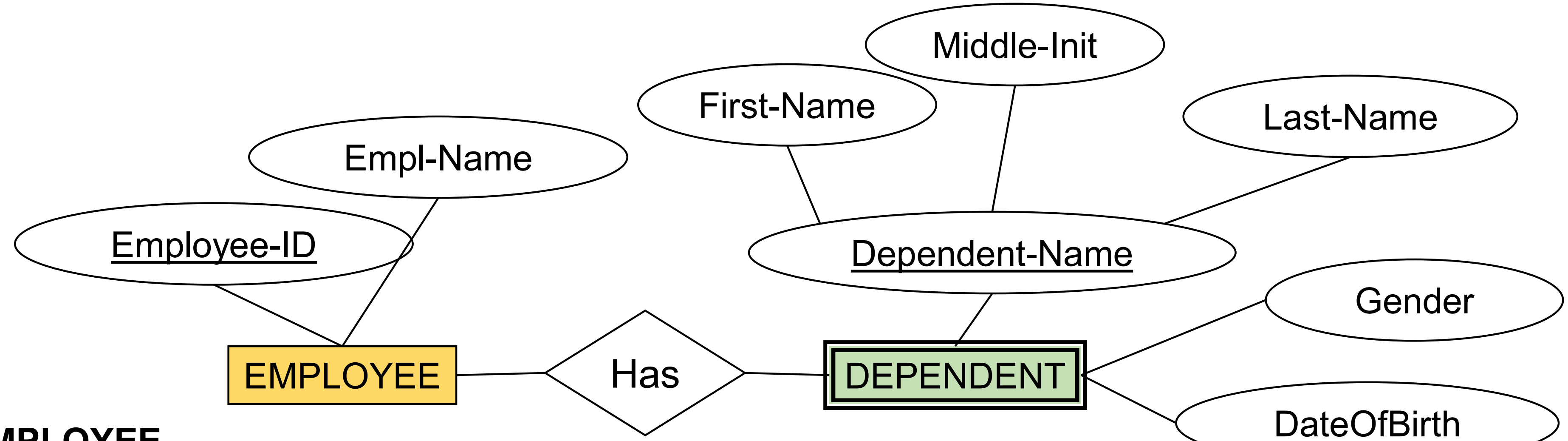
EMPLOYEE-SKILL

Employee-ID	Skill

2. Mapping Weak Entities

- For each weak entity create a new relation
  - include all the simple attributes
  - include the primary key of the owner relation as a foreign key attribute in this new relation
  - the primary key of this new relation is a combination of the primary key of the owner and the partial identifier of the weak entity

Mapping Weak Entities



EMPLOYEE

Employee-ID	Empl-Name

DEPENDENT

First-Name	Middle-Init	Last-Name	Employee-ID

DateOfBirth	Gender

NOTE: the domain constraint for the foreign key should NOT allow ***null*** value if DEPENDENT is a weak entity

Weak entitynin existenc strong entitye baldr bu yüzden eer strong entity'den gelen foreign key null olursa weak entity exist olamaz bu yüzden domain constraint for the foreign key should not allow null value.

3. Mapping Binary Relationships

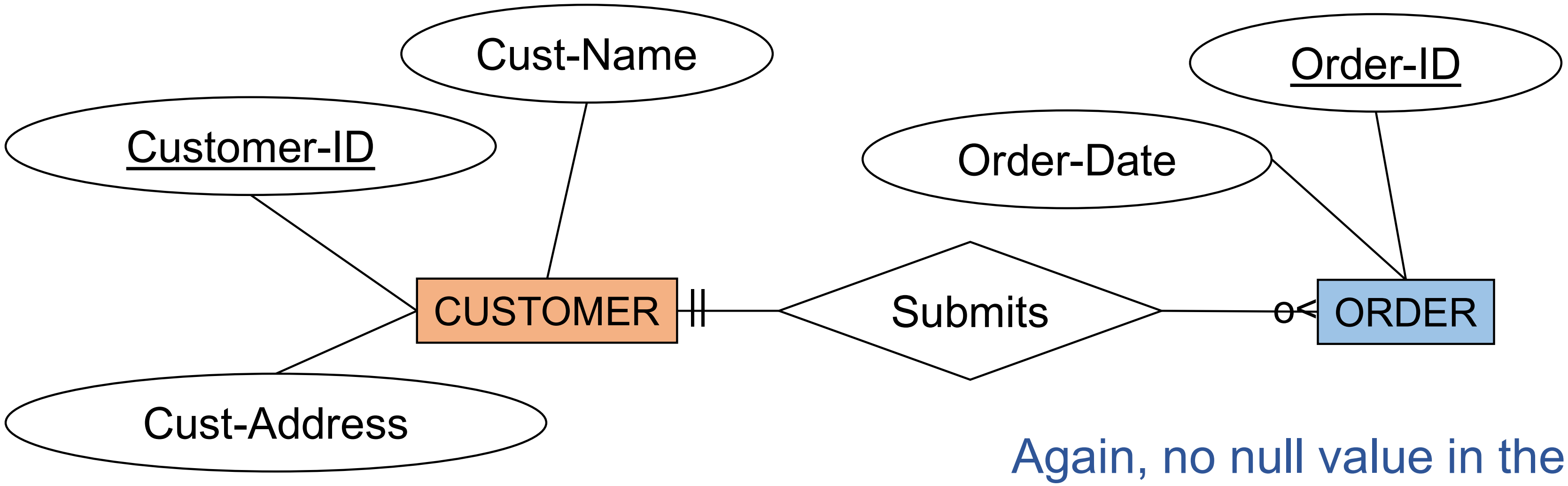
• One-to-one	1:1
• One-to-many	1:M
• Many-to-many	M:N

Mapping Binary Relationships 1:M

- Create a relation for each of the two entities
  - include the primary key of the “one” side as a foreign key in the “many” side relation
  - “The primary key migrates to the many side.”

1- many -> 1 olan tarafn keyi her zaman many olan tarafta tutulur.
many to many-> her iki relationdaki keyler tutulur.

1:M Binary Relationship



Again, no null value in the foreign key…this is because of the mandatory minimum cardinality

CUSTOMER

Customer-ID	Cust-Name	Customer-Address

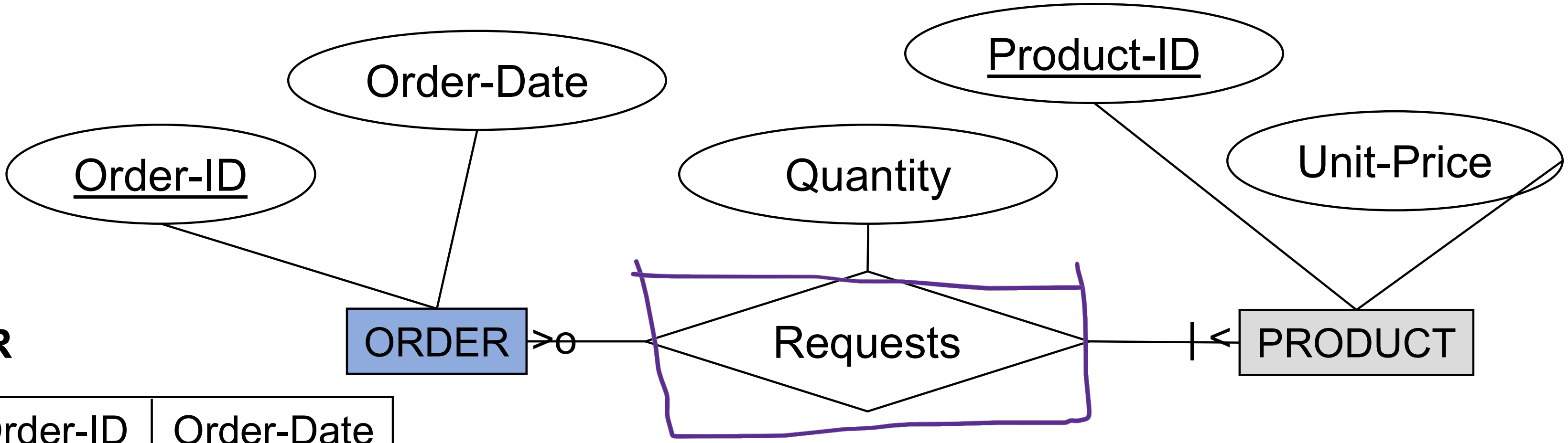
ORDER

Order-ID	Order-Date	Customer-ID

Mapping Binary M:N

- Create three relations
  - one for each entity
  - one that includes the primary key of each entity and any other attributes that are unique to the relationship

Mapping M:N Binary Relationship



ORDER

Order-ID	Order-Date

ORDER-LINE

Order-ID	Product-ID	Quantity

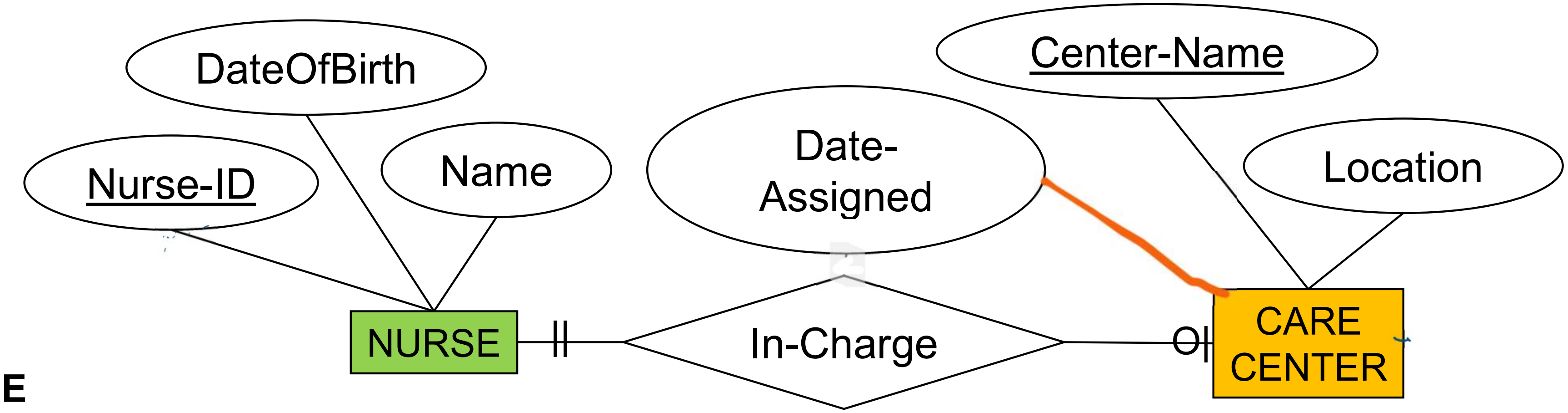
PRODUCT

Product-ID	Unit-Price

Mapping Binary 1:1 Relationship

- 1:1 is a special case where the primary key of one relation is a foreign key of the other
- If you have mandatory and optional cardinalities
  - the optional side should get the primary key of the mandatory side as a foreign key

Mapping Binary 1:1



NURSE

Nurse-ID	Name	DateOfBirth

CARE CENTER

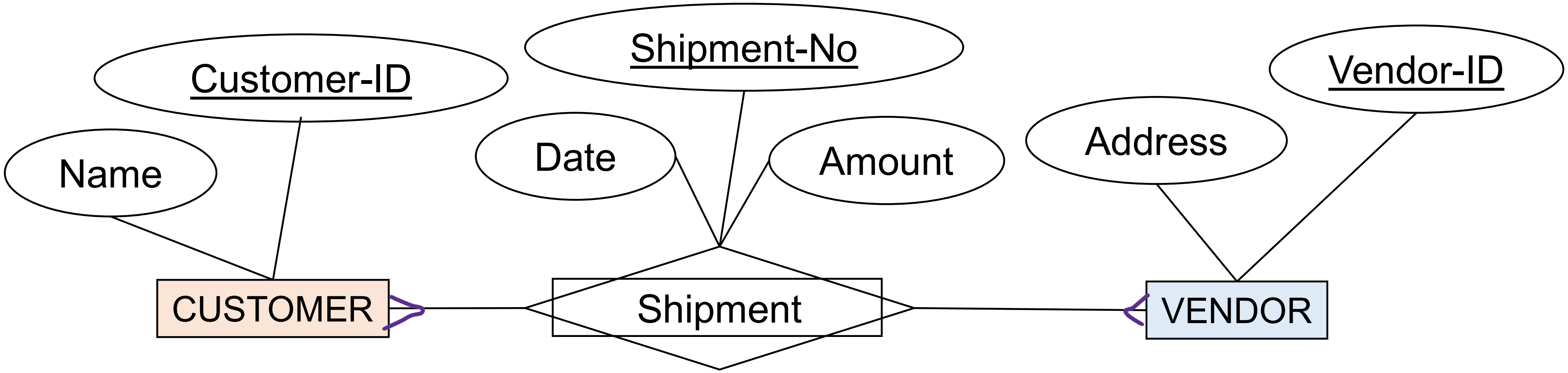
Center-Name	Location	Nurse-In-Charge	Date-Assigned

Optional Participant

4. Mapping Associative Entities

- Identifier not assigned:
  - Same as Binary M:N when Associative entity does not have a unique Identifier
- Identifier assigned:
  - Include it as primary key in third relation and add the primary key of the other two relations as foreign keys

Associative Entities w/Identifier



CUSTOMER

Customer-ID	Name

SHIPMENT

Shipment-No	Customer-ID	Vendor-ID	Date	Amount

VENDOR

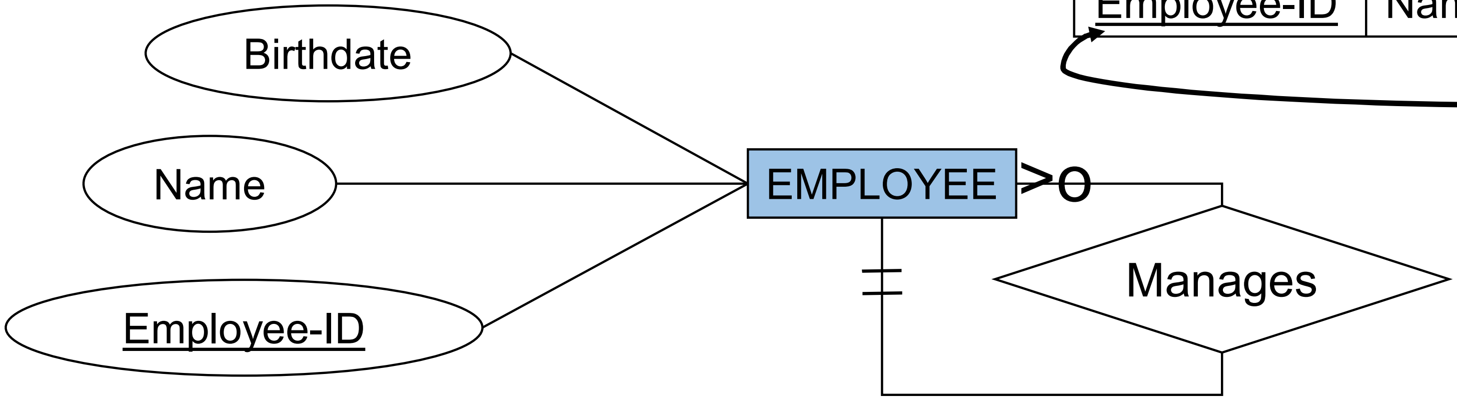
Vendor-ID	Address

5. Mapping Unary 1:M Relationships

- Create one relation
- Add a foreign key within the same relation that references the primary key

EMPLOYEE

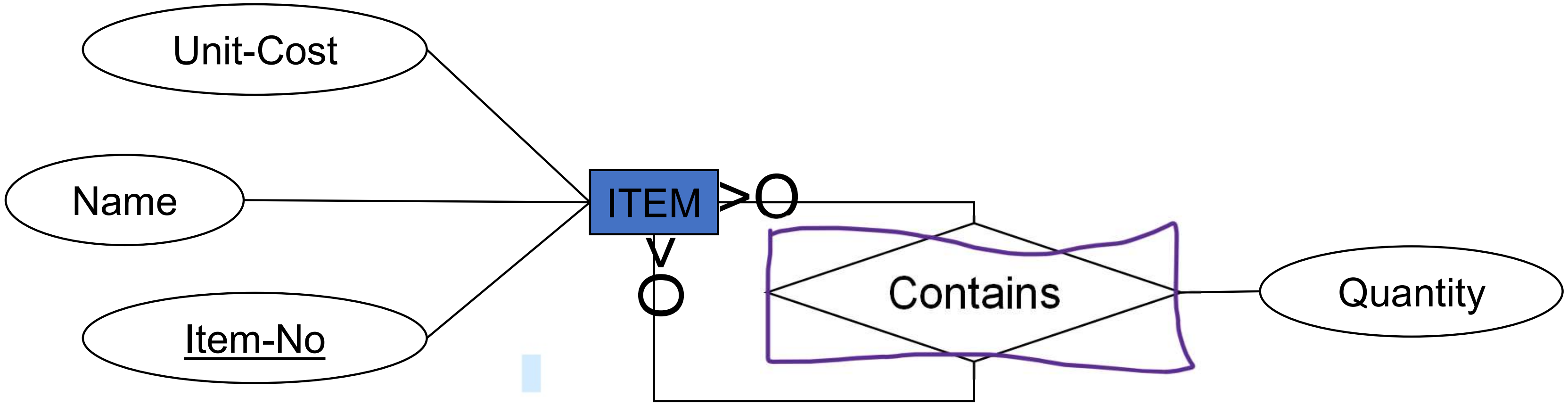
Employee-ID	Name	Birthdate	Manager-ID



Mapping Unary M:N Relationships

- Create two relations
  - One to represent the entity, the other represents the M:N relationship
  - The primary key of the association relation are two attributes that take their values from the primary key of the original entity

Mapping M:N Unary



ITEM

<u>Item-No</u>	Name	Unit-Cost

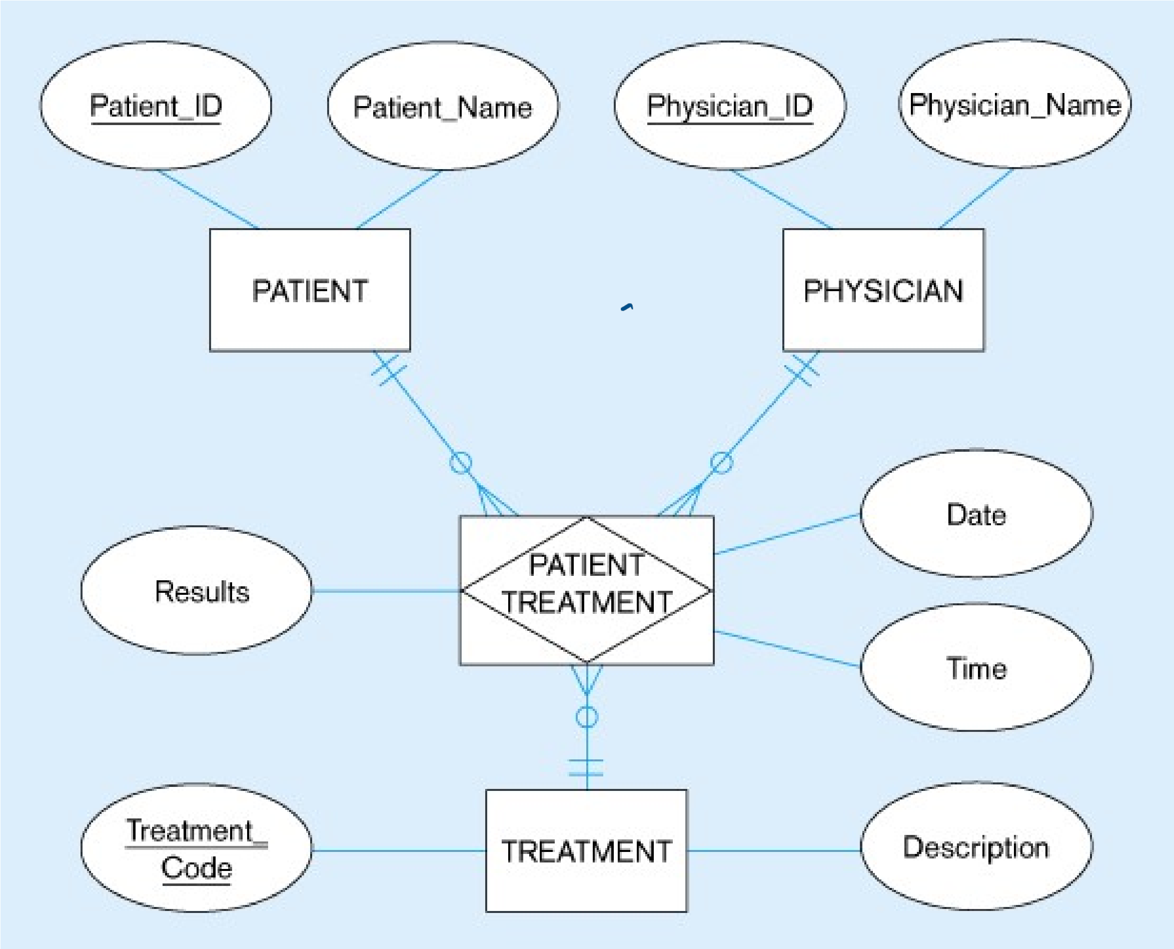
COMPONENT

<u>Item-No</u>	<u>Component-No</u>	Quantity

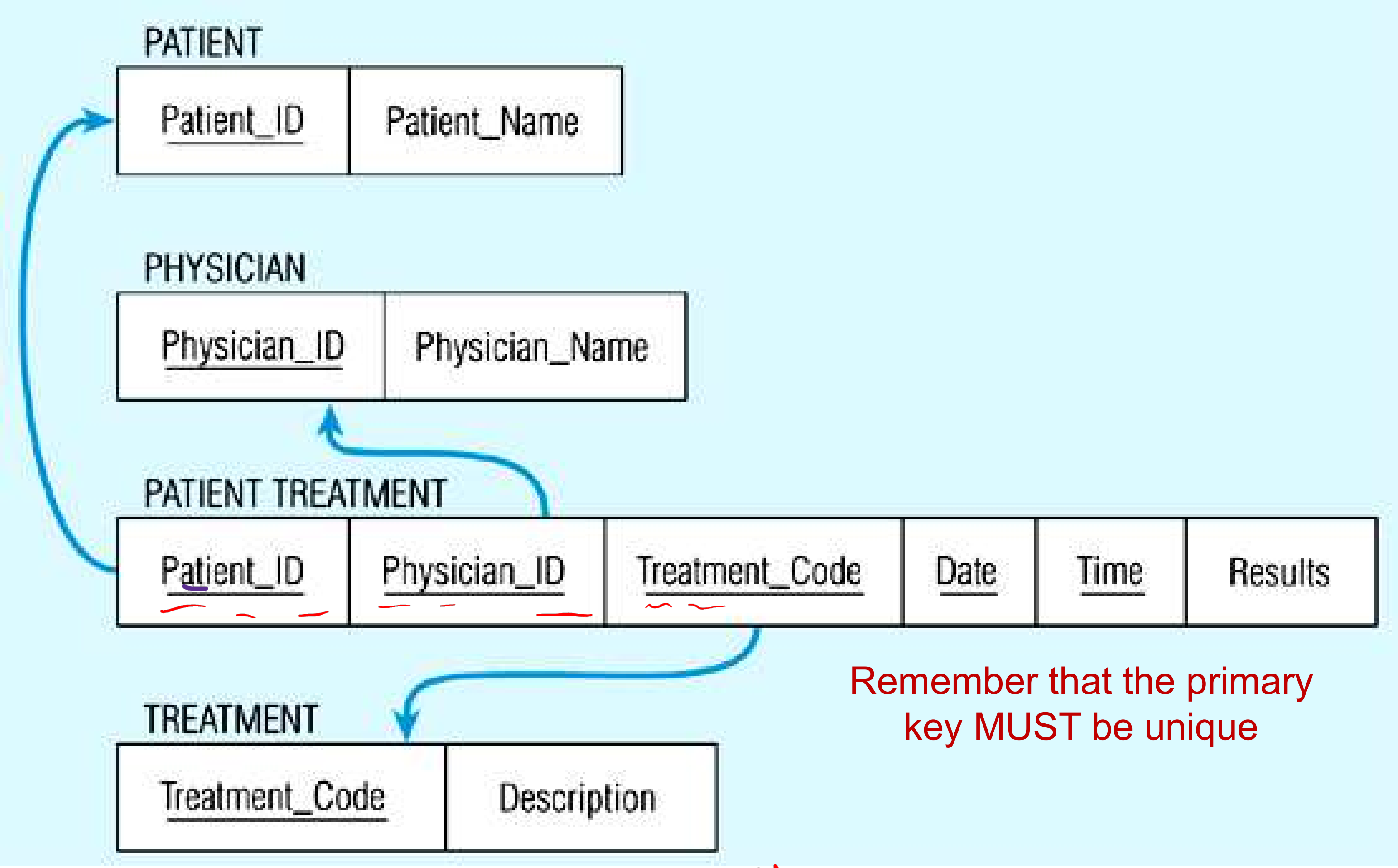
6. Mapping Ternary Relationships

- Create four relations, one for each entity and an additional relation to represent the relationship
  - The primary key of the four relation is made up of the primary keys of each of the entities and additional attributes needed to uniquely identify an instance

Mapping a ternary relationship
Ternary relationship with associative entity



Mapping the ternary relationship



PATIENT

Patient_ID	Patient_Name

PHYSICIAN

Physician_ID	Physician_Name

PATIENT TREATMENT

Patient_ID	Physician_ID	Treatment_Code	Date	Time	Results

TREATMENT

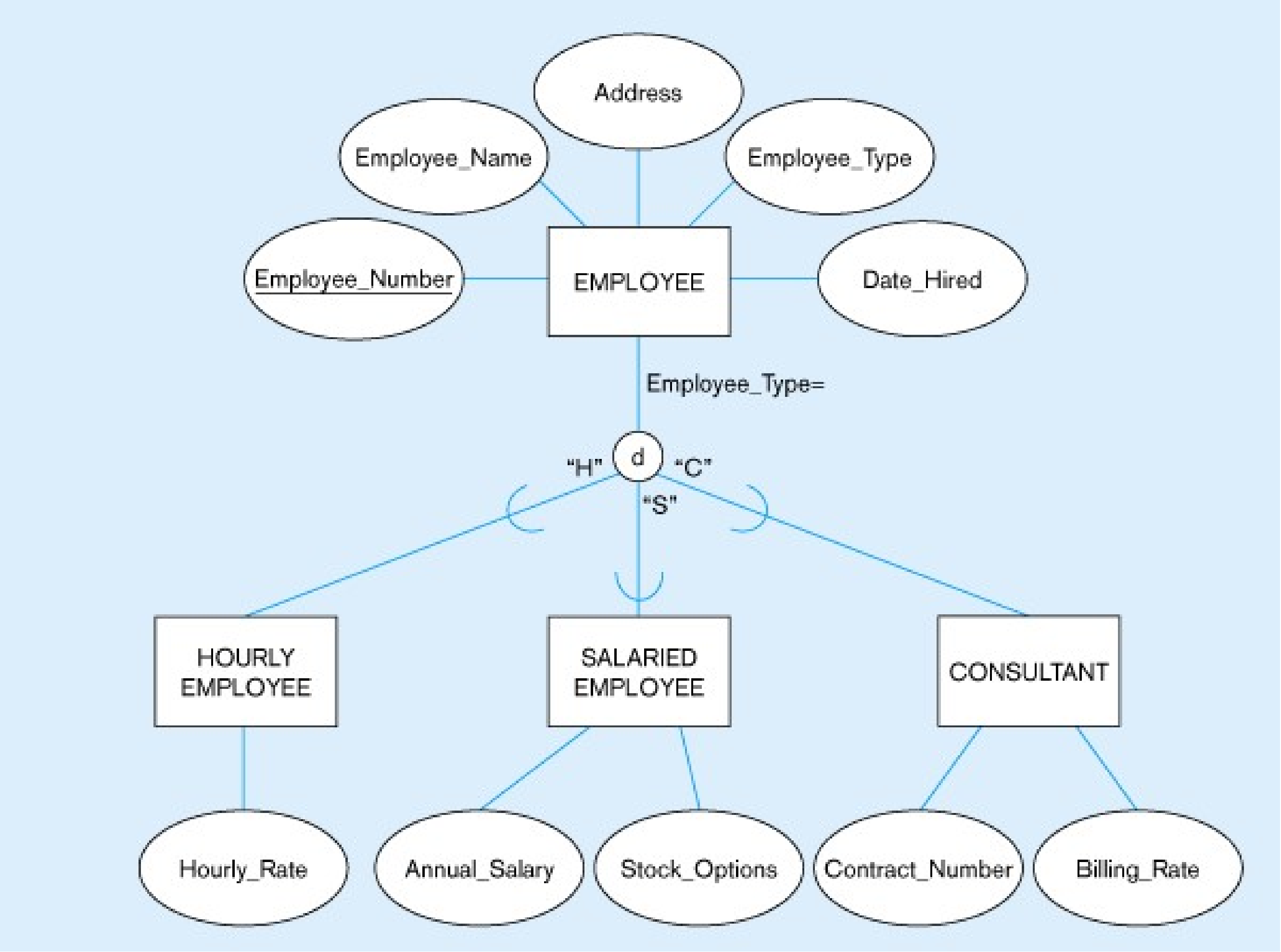
Treatment_Code	Description

Remember that the primary key MUST be unique

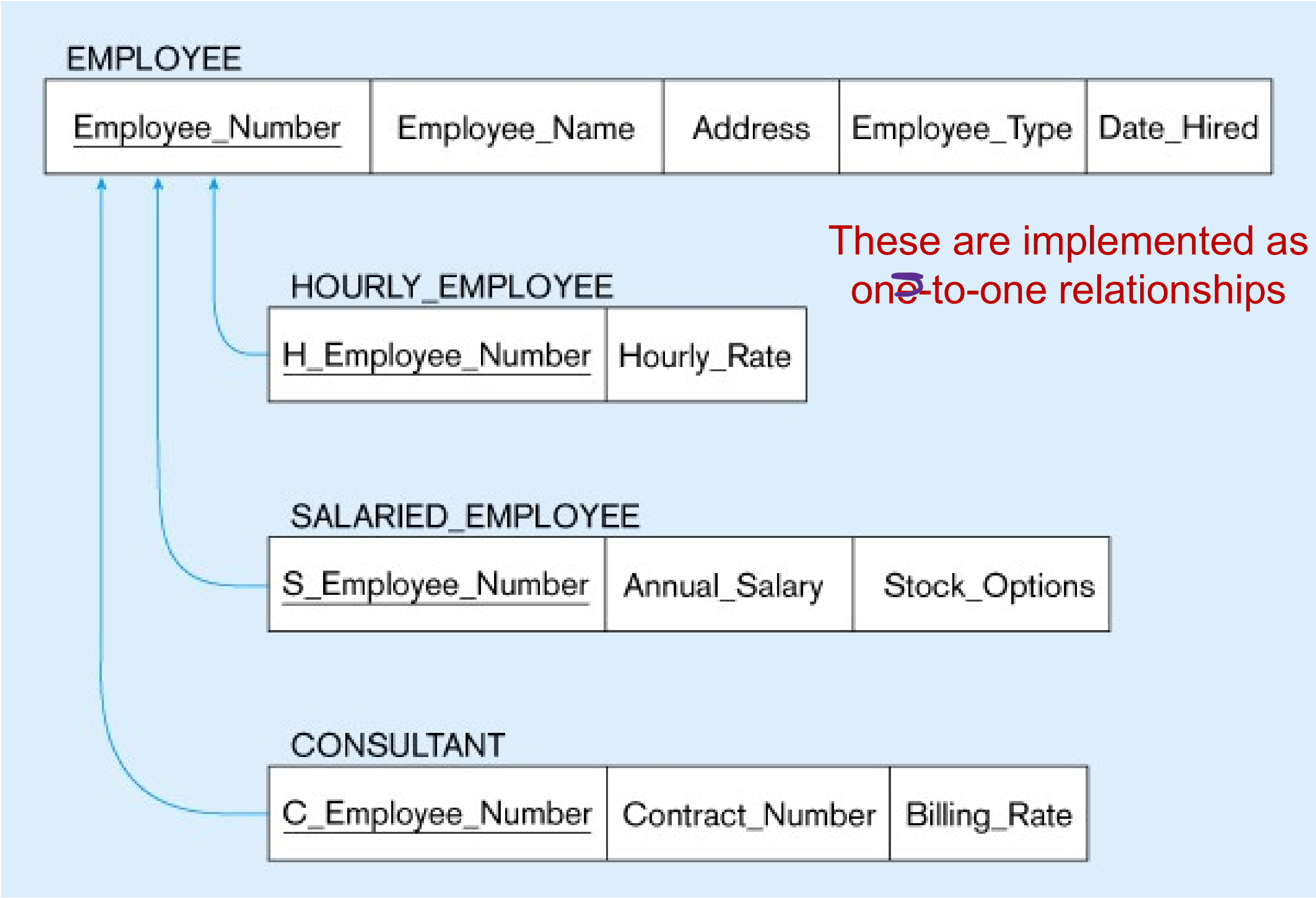
7. Mapping Super/Subtypes

- Create a separate relation for the supertype and each of its subtypes
- Assign to supertype relation the attributes common to all members of the supertype, including the primary key
- Assign to the subtype relations the primary key of the supertype and the attributes unique to that subtype
- Assign one(or more) attributes of the supertype to function as the subtype discriminator.

Supertype/subtype relationships



Mapping Supertype/subtype relationships to relations



EMPLOYEE

Employee_Number	Employee_Name	Address	Employee_Type	Date_Hired

HOURLY_EMPLOYEE

H_Employee_Number	Hourly_Rate

SALARIED_EMPLOYEE

S_Employee_Number	Annual_Salary	Stock_Options

CONSULTANT

C_Employee_Number	Contract_Number	Billing_Rate

These are implemented as one-to-one relationships

Normalization

- The process of converting complex data structures to simple stable structures
- Normal Form: A state of a relation that can be determined by applying simple rules regarding dependencies to that relation
- Primarily a tool to validate and improve a logical design so that it satisfies certain constraints that ***avoid unnecessary duplication of data***
- The process of decomposing relations with anomalies to produce smaller, ***well-structured*** relations

Complex data structurelarm decompose edip içindeki unnecessary duplication of datalardan kurtarp anomalileri yok etmek.

Well-Structured Relations

- A relation that contains minimal data redundancy and allows users to insert, delete, and update rows without causing data inconsistencies
- Goal is to avoid anomalies
  - **Insertion Anomaly** – adding new rows forces user to create duplicate data
  - **Deletion Anomaly** – deleting rows may cause a loss of data that would be needed for other future rows
  - **Modification Anomaly** – changing data in a row forces changes to other rows because of duplication